Т. 5. №7. 2019
DOI: 10.33619/2414-2948/44

УДК 004.45 https://doi.org/10.33619/2414-2948/44/20

ОЦЕНКА ЗАЩИЩЕННОСТИ СРЕДЫ ВЫПОЛНЕНИЯ ОПЕРАЦИОННОЙ СИСТЕМЫ ANDROID

©Воронин А. А., *ORCID: 0000-0003-4375-7559, SPIN-код: 2892-7023, канд. техн. наук, Владимирский государственный университет им. А.Г. и Н.Г. Столетовых, г. Владимир, Россия, a_voron@mail.ru*
©Виноградов Д. В., *ORCID: 0000-0001-5366-6461, SPIN-код: 8253-2070, Владимирский государственный университет им. А.Г. и Н.Г. Столетовых, г. Владимир, Россия, dvv@eup.ru*
©Воронина А. А., *ORCID: 0000-0003-0386-021X, SPIN-код: 8290-2857, Владимирский государственный университет им. А.Г. и Н.Г. Столетовых, г. Владимир, Россия, xannet@mail.ru*

SECURITY ESTIMATION OF ANDROID EXECUTION ENVIRONMENT

©Voronin A., *ORCID: 0000-0003-4375-7559, SPIN-code: 2892-7023, Ph.D., Vladimir State University, Vladimir, Russia, a_voron@mail.ru*
©Vinogradov D., *ORCID: 0000-0001-5366-6461, SPIN-код: 8253-2070, Vladimir State University, Vladimir, Russia, dvv@eup.ru*
©Voronina A., *ORCID: 0000-0003-0386-021X, SPIN-code: 8290-2857, Vladimir State University, Vladimir, Russia, xannet@mail.ru*

Аннотация. В работе рассмотрены основные проблемы информационной безопасности среды выполнения операционной системы Android, сформулированы требования к системе аудита среды выполнения. В ходе экспериментальных исследований произведена оценка 2178 уязвимостей ОС, а также проанализированы 659 приложений на 20 мобильных устройствах на наличие дефектов безопасности. Результаты продемонстрировали критически низкий уровень защиты среды выполнения ОС на всех тестируемых устройствах.

Abstract. The paper considers main problems of information security of Android OS execution environment and defines basic requirments to execution ebvironment audit system. During experimental studies assessment of 2178 OS vulnerability assessment was made, also were analized 659 applications of 20 mobile devices for the presence of security defects. The results have showed a critically low protection level of OS execution environment on all tested devices.

Ключевые слова: среда выполнения, android, уязвимость.

Keywords: execution environment, android, exploit.

Среда выполнения операционной системы является одним из ключевых элементов любой операционной системы, предоставляя возможность выполнения приложений, независимо от языка программирования на котором они были реализованы.

Каждое приложения в ОС Android использует свою среду выполнения (СВ ОС), загружаемую в память устройства в процессе запуска приложения. СВ ОС осуществлвяет подготовку к выполнению и выполнение приложения на устройстве, в ходе которой может осуществляться оптимизация и компиляция кода. Также к функциям СВ ОС относится установка приложения на устройство. По мере развития операционной системы Android менялась архитектура СВ ОС:

Т. 5. №7. 2019
DOI: 10.33619/2414-2948/44

- в ОС до версии 2.1 использовалась высокоуровневая ВМ, интерпретирующая байт-код (Dalvik);

- в ОС до версии 4.4 использовалась высокоуровневая ВМ, интерпретирующая байт-код с одновременным использованием JIT компиляции (Dalvik);

- в ОС до версии 6 использовалась низкоуровневая ВМ и набор подключаемых библиотек с полной OAT компиляций байт-кода (ART);

- в современных версиях ОС примененяется низкоуровневая ВМ и набор подключаемых библиотек с частичной OAT компиляцией байт-кода с применением JIT-компилятора (ART).

При этом сам процесс загрузки среды выполнения остается неизменным.

На состояние безопасности СВ ОС существенное влияние оказывают особенности мобильных устройств на которых используется ОС Android [1], такие как миниатюрность, мобильность, ограниченность вычислительных ресурсов, а также вносимые производителями мобильных устройств модификации операционной системы.

Атаки на СВ ОС реализуются через уязвимости пользовательских приложений и их внешних компонентов, которые в основном вызваны низким качеством разработки и тестирования мобильных продуктов [2], а именно: некорректным использованием платформы, небезопасным хранением и передачей данных, небезопасной аутентификаций и авторизаций, использованием алгоритмов с низкой криптографической стойкостью, низким качеством, наличием скрытых функций. Множество представленных уязвимостей рассмотрены в работах [3, 4].

Для реализации угроз в ряде случаев даже не нужен доступ с уровнем администратора. Такие операции как установка apk файлов, содержащих вредоносный код, переупаковка легальных apk файлов, обход подписей приложений и использовани избыточных разрешений могут быть выполнены с использованием полномочий рядового пользователя.

В СВ ОС регулярно выявляются ошибки. Например, с среди наибоее критичных уязвимостей можно выделить:

- уязвимость CVE-2017-0780 [5], которая позволяет злоумышленнику вызывать зависание приложения при его выполнении с помощью специально созданного файла;

- уязвимость CVE-2017-13309 [6], которая позволяет получить доступ к данным, открытым только для установленных на устройстве приложений, которые обладают необходимыми разрешениями;

- уязвимость CVE-2017-13176[7] и CVE-2017-13274 [8], которые позволяют обойти требования к взаимодействию с пользователем и получить доступ к дополнительным разрешениям;

- уязвимость CVE-2016-6703 [9], которая позволяет выполнять произвольный код в контексте непривилегированного процесса с помощью специально созданных данных).

В ОС Android реализована многоуровневая система безопасности, реализующая принцип предоставления минимальных прав. Защита СВ ОС дополнена технологиями DEP (предотвращение выполнения данных, Data Execution Prevention) и ASLR (предотвращение выполнения кода из неисполняемых областей памяти, Address space layout randomization).

Основные процессы СВ ОС, возможные угрозы и механизмы защиты от них приведставлены в Таблице 1.

Под аудитом информационной безопасности будем понимать оценку текущего состояния системы информационной безопасности, устанавливающую уровень ее соответствия определенным требованиям и ограничениям.

Т. 5. №7. 2019
DOI: 10.33619/2414-2948/44

Таблица 1.

ОСНОВНЫЕ МЕХАНИЗМЫ ЗАЩИТЫ СВ ОС

Процесс	*Угрозы*	*Механизм защиты*
Загрузка СВ	Требуемый уровень доступа «root»: форсирование применение механизма fork; изменение загрузочных образов среды выполнения	запрет работы с правами root, ASLR, контроль загрузки компонентов ОС, контроль целостности образов
Установка и обновление приложений	Требуемый уровень доступа «root»: изменение содержимого легальных файлой после распаковки; перезапись файла platform.xml Требуемый уровень доступа «Пользователь»: установка файлов, содержащих вредоносный код; изменение и переупаковка легальных файлов; обход подписи приложений; запрет установки приложений	песочница, проверка подписи приложений, проверка на вирусы, статический и динамический анализ кода, Google Play Protect
Исполнение приложений	Требуемый уровень доступа «root»: замена или внедрение вредоносного кода в исполняемые файлы приложения Требуемый уровень доступа «Пользователь»: форсирование использования ресурсов; осуществление сговора, использование уязвимостей или избыточных разрешений легальных приложений	песочница, контроль разрешений, Binder IPC, блокировка опасных системных вызовов, DEP, ASLR, контроль целостности исполняемых файлов и загрузочных образов

Основная цель проведения аудита — поиск возможных или реальных нарушений безопасности.

Аудит може осуществляться в режиме реального времени и в пакетном режиме.

Основные этапы аудита и их взаимосвязь приведены на Рисунке 1.

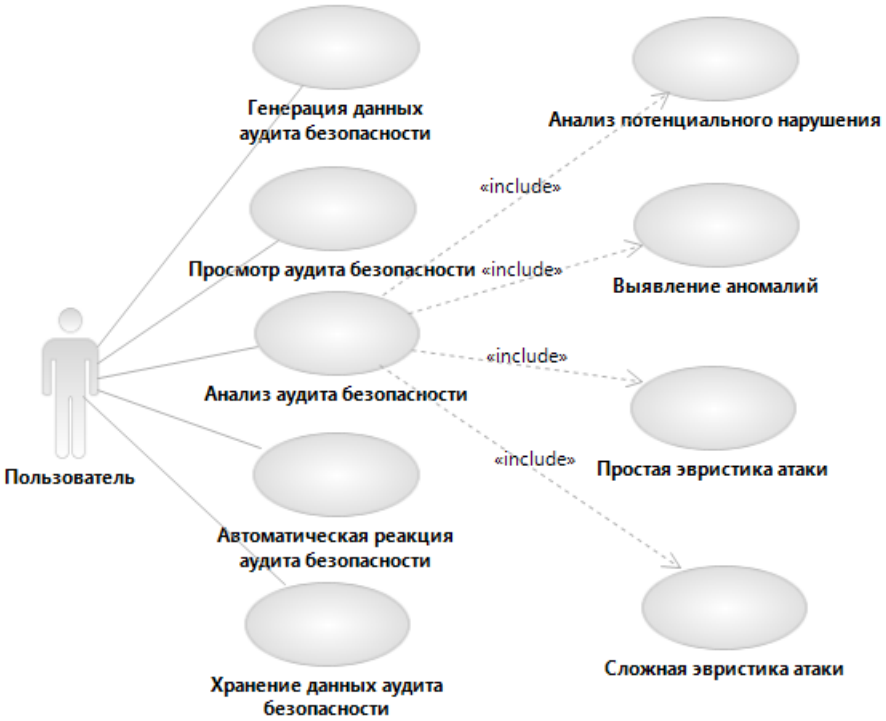

Рисунок 1. Элементы аудита информационной безопасности

Использование механизмов защиты ОС позволяет минимизировать риск проведения известных атак на СВ ОС, но не исключет выявление новых уязвимостей.

Уязвимости могут быть выявлены путем сравнения конфигурации установленной ОС с ее эталонным образцом. Данный подход не требует root-доступа к устройству, однако при

Т. 5. №7. 2019
DOI: 10.33619/2414-2948/44

взломе мобильного устройства становится неэффективным, так как информации о характеристиках ОС в данном случае может быть изменена.

Уязвимости, связанные с переупаковкой приложений могут быть выявлены путем выполенения приложений в песочнице. Характеристики приложения, после его установки в песочнице, могут сравниваться с эталоном, хранящегося в защищенном месте (например, в Google Play или в публичном или частном депозитарии). В данном случае возможно использование полного побитового сравнения, обнаружение расхождений на основе нечеткого хеширования, обнаружение расхождений на основе графа программных зависимостей, обнаружение расхождений на основе хеширования функций, обнаружение расхождений на основе динамически внедренного кода, обнаружение расхождений на основе использования водяных знаков.

Уязвимости, связанные с внедрением вредоносного кода могут быть выявлены методами, основанными на идентификации злоупотреблений (аномалий). Методы, основанные на выявлении аномалий, предполагают разработку моделей легитимной функциональности [10]. Однако такой подход является затруднительным в реализации, так как требует учета множества шаблонов нормального состояния и поведения приложения, которые в основном разрабатываются вручную [11].

Уязвимости, связанные с привилегиями могут быть выявлены путем анализа информации из манифеста приложений. Данный подход предполагает:

- сопоставление конфигурации анализируемого приложения с множеством предопределенных правил [12], описывающих «опасные» конфигурации привилегий объявленных в манифесте приложения;
- проверку соответствия запрашиваемых привилегий предоставленному описанию приложения [13];
- сопоставление запрашиваемых привилегий с привилегиями известных вредоносных приложений, поиск характерных для вредоносных программ API-вызовов [14].

Аудит может быть основан на внедрении специального кода в память процесса приложения. При этом следует учитывать, что внедрение кода с одинаковым успехом может быть использовано как для мониторинга работы СВ ОС, так и для атаки на нее.

Также необходимо выявлять уязвимости, связанные с нерегулярным обновлением ОС.

Характеристика использования ресурсов СВ ОС может быть получена с использованием средств разработки, например, инструментария Android Debug Bridge (adb).

Автоматизация аудита СВ СО сводится к управлению конфигурацией СВ ОС, позволяющей согласовать ее состояние с требованиями политики безопасности организации.

В большинстве подходов для проведения аудита необходимо обладать правами суперпользователя (root доступ). Следует учитывать, что при реализации программных средств аудита наличие у этих приложений прав суперпользователя, может создать дополнительные уязвимости. Поэтому система аудита должна решать большую часть поставленных задач (оптимально — все задачи) без предоставления root-доступа к ОС, т.е. с правами гостевого приложения или правами администратора конфигурации.

Наиболее эффективными и быстро реализуемыми механизмами аудита в данном случае могут быть следующие: проверка на наличие root-доступа к ОС; поиск и оценка известных уязвимостей СВ ОС; выявление и оценка возможных злоупотреблений (аномалий); контроль целостности используемых файлов.

В работе аудит СВ ОС проводился с помощью разработанного приложения. Приложение имеет клиент-серверную архитектуру (Рисунок 2):

Т. 5. №7. 2019
DOI: 10.33619/2414-2948/44

- Мобильное устройство осуществляет сбор данных и вывод результатов аудита. Основной режим работы клиентского приложения — фоновый. При получения рекомендаций, выводится всплывающее уведомление. Полный отчет по результатам аудита доступен на сервере.

- Сервер осуществляет обработку собранных данных, формирование рекомендаций по результатам аудита. Обработка данных и формирование отчетов осуществляется агентами, которые запускаются по рассписанию. Доступ к отчетам других устройств невозможен.

Благодаря такой архитектуре разработанное ПО может быть использовано в организациях и на предприятиях в рамках MDM-решений (Mobile Device Management).

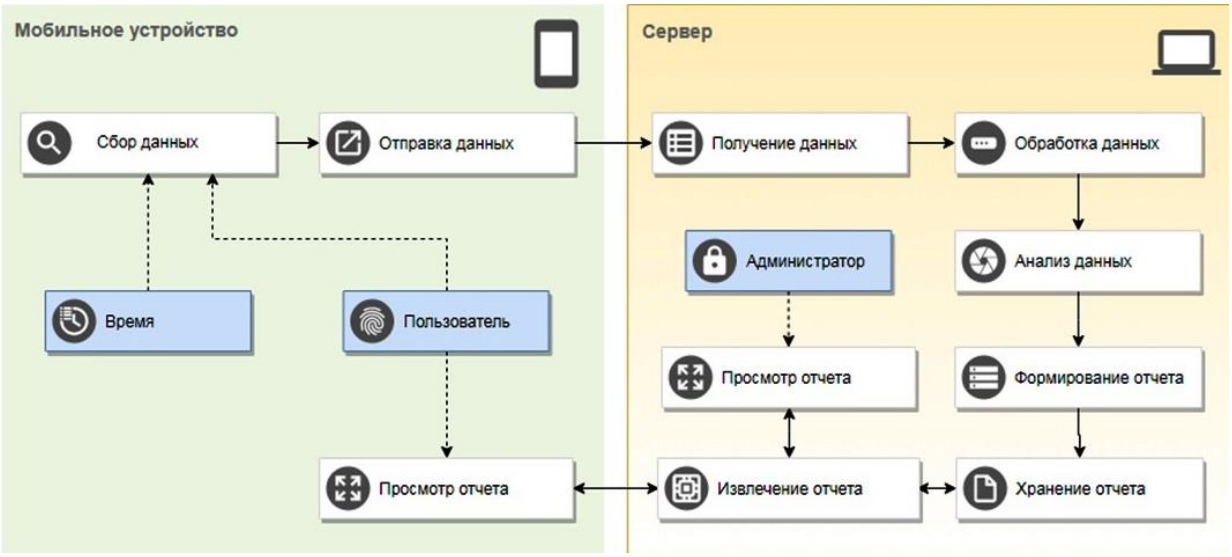

Рисунок 2. Архитектура приложения автоматизированного аудита

Алгоритм проведения аудита предполагает:

- Получение оценки защищенности. При расчете оценки защищенности используются параметры конфигурации устройства, которая зависит от производителя устройства, типа используемой платформы, версии установленной ОС, а также последнего установленного обновления ОС. Для описания конфигурации может быть использован перечень общеизвестных платформ (Common Platform Enumeration, CPE) и перечень общеизвестных конфигураций (Common Configuration Enumeration, CCE). Для описания уязвимостей могут быть использованы сведения, содержащиеся в Common Vulnerabilities and Exposures (CVE), Common Weakness Enumeration (CWE), а также публикуемые ФСТЭК РФ сведения об уязвимостях. Результат оценки уровня защищенности — выдача предупреждения пользователю (администратору) с указанием на необходимость проведения обновления ОС, в том случае если значение результирующего показателя будет положительным.

- Выявления дефектов безопасности приложений (сигнатурный анализ). Целью проверки является выявление дефектов безопасности (в т. ч. преднамеренного характера). Признаками возможного присутствия дефектов безопасности в приложениях, как правило, являются: обращение к запрещенным объектам; использование устаревших функций и алгоритмов; ошибки разработчика, например, хранение временных файлов в общедоступном месте, ведение журнала событий, отказ от использования конфигурации сетевой безопасности, использование жестко прописанных в коде и конфигурационных файлах паролей. Результат оценки - сообщение об уровне опасности устанавливаемого приложения.

- Выявления дефектов безопасности приложений (анализ привилегий). Целью проверки является выявление дефектов безопасности (т. ч. преднамеренного характера), которые

Т. 5. №7. 2019
DOI: 10.33619/2414-2948/44

связаны с запрашиваемыми привилегиями. Данные о о запрашиваемых привилегиях содержатся в файле AndroidManifest.xml проверяемого приложения (при наличии прав суперпользователя – packages.xml, platform.xml). Признаками возможного присутствия в клиентских приложениях дефектов безопасности, как правило, являются запрашиваемые приложением привилегии, которые являются избыточными для реализации его функций, отраженных в документации. Результат оценки - сообщение об уровне опасности устанавливаемого приложения, с точки зрения запрашиваемых привилегий.

- Проверка целостности загрузочных образов. Целью проверки является выявление подделки загрузочных образов СВ ОС и приложений. В процессе контроля необходимо осуществлять проверку целостности файлов, расположенных в каталоге для временного хранения файлов библиотек СВ ОС («/data/dalvik-cache»), файлов, расположенных каталоге исполняемых файлов приложения и образов в памяти процесса приложения. Проверки подлежат файлы .odex, .vdex, .art, .oat, .dex, размещенные на мобильном устройстве, и их образы в его оперативной памяти. Источником информации об эталонных файлах могут быть файлы, размещенные на эталонном эмуляторе устройств различных платформ, таких как Mips, Mips64, X86, X86 64, Arm, Arm64. Необходимо учитывать, что некоторые файлов является условно неизменяемыми (их изменение обычно связано либо с обновлением системы, либо обновлением приложения). Другая часть файлов меняется регулярно. Изменения вызваны оптимизацией приложений и обновлениями компонентов ОС. Образы файлов в оперативной памяти могут быть получены и проанализированы с использованием интерфейса «/proc/self/maps» (при наличии root-доступа). Результат оценки — выдается сообщение о выявленных расхождениях.

В ходе исследований произведена оценка 2178 выявленных в ОС уязвимостей по состоянию на июнь 2019 года. Проанализированы 659 приложений на 20 мобильных устройствах.

В результате экспериментальных исследований продемонстрировали критически низкий уровень защиты СВ ОС на всех тестируемых устройствах — 12 проверенных устройств имели показатель на уровне -критический риск, остальные — высокий риск.

Показательные результаты дала проверка на наличие дефектов: сигнатурный анализ и анализ привилегий.

Сигнатурный анализ показал, что критический уровень имеет 107 приложений (16,23% от общего количества), высокий — 226 (34.29%), средний — 306 (46,23%), низкий — 20 (3.03%). На Рисунке 3 представлены данные по выявленным уязвимостям в разрезе ключевых показателей информационной безопасности.

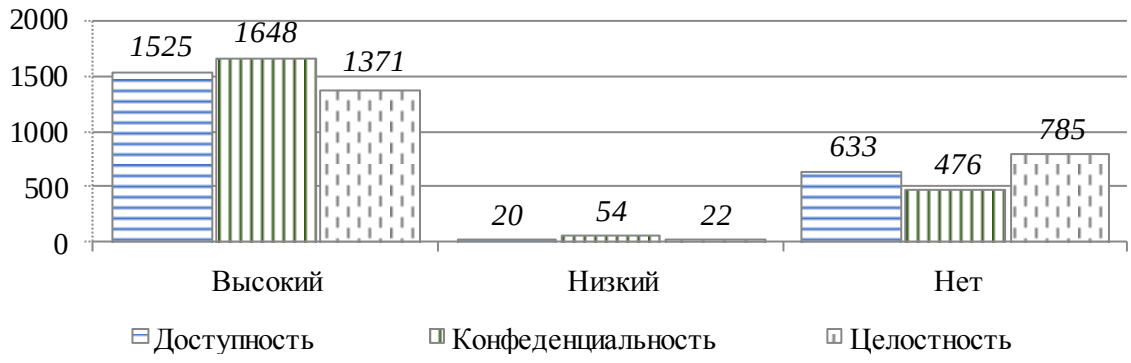

Рисунок 3. Распределение уязвимостей в разрезе ключевых показателей ИБ

Т. 5. №7. 2019
DOI: 10.33619/2414-2948/44

В результате анализа разрешений было проанализировано было выявлено, что критический уровень имеет 10 приложений (1,51% от общего количества), высокий — 240 (36.4%), средний — 297 (45%), низкий — 112 (16.9%).

Результаты совместного анализ дефектов ИБ приложений с применением метода сигнатурного анализа и метода анализа разрешений представлены в Таблице 2.

Таблица 2.

РЕЗУЛЬТАТЫ СВОДНОГО АНАЛИЗА ВЫЯВЛЕНИЯ ДЕФЕКТОВ

Уровень опасности (анализ разрешений)	*Уровень опасности (сигнатурный анализ)*			
	Критический	*Высокий*	*Средний*	*Низкий*
Критический	2	50	48	7
Высокий	6	121	84	15
Средний	2	67	155	82
Низкий	0	2	10	8

Результаты исследования свидетельствуют о низком уровне безопасности среды выполнения ОС. Таким образом представляется актуальным разработка методов автоматического аудита СВ ОС. В организациях целесообразно усиление контроля за использованием сотрудниками мобильных устройств на базе ОС Android, ввиду наличия большого количества уязвимостей, которые могут быть использованы злоумышленником для получения доступа к конфиденциальной информации.

Список литературы:

1. Маркин Д. О., Комашинский В. В., Баранов И. Ю. Модель управления профилем защиты мобильного устройства при доступе к услугам с разным уровнем конфиденциальности // Информационные технологии. 2015. 21(8). С. 611-618.

2. Mobile Top 10 2016-Top 10 https://clck.ru/Gsqgp.

3. Зубков К. Н., Диасамидзе С. В. Проблемы защиты информации в приложениях для мобильных систем // Intellectual Technologies on Transport. 2017. №2. С. 40-46.

4. Lu L., Li Z., Wu Z., Lee W., Jiang G. Chex: statically vetting android apps for component hijacking vulnerabilities // Proceedings of the 2012 ACM conference on Computer and communications security. ACM. 2012. P. 229-240. DOI: 10.1145/2382196.2382223

5. Бюллетень по безопасности Android – сентябрь 2017 г. https://clck.ru/GsqhF

6. Бюллетень по безопасности Android – май 2018 г: https://clck.ru/GsqhQ

7. Бюллетень по безопасности Android – январь 2018 г. https://clck.ru/Gsqhg

8. Бюллетень по безопасности Android – апрель 2018 г. https://clck.ru/Gsqhx

9. Бюллетень по безопасности Android – ноябрь 2016 г. https://clck.ru/GsqiK

10. Shabtai A., Tenenboim-Chekina L., Mimran D., et al. Mobile Malware Detection through Analysis of Deviations in Application Network Behavior. // Computers & Security. 2014. 43. P. 1-18. DOI:10.1016/j.cose.2014.02.009

11. Александров Я. А., Сафин Л. К., Трошина К. Н., Чернов А. В. Статический бинарный анализ мобильных приложений для платформы Android по требованиям информационной безопасности // Вестник Московского университета. Серия 15. Вычислительная математика и кибернетика. 2016. №3. С. 44-49.

12. Enck W., Ongtang M., McDaniel P. On Lightweight Mobile Phone Application Certification // Proceedings of the 16th ACM Conference on Computer and Communications Security. NY, USA: ACM. 2009. P. 235-245. DOI:10.1145/1653662.1653691

13. Цыганенко Н. П. Статический анализ кода мобильных приложений как средство выявления его уязвимостей // Труды БГТУ. Серия 3: Физико-математические науки и информатика. 2015. №6 (179). С. 200-203.

14. Skovoroda A., Gamayunov D. Securing mobile devices: malware mitigation methods // JoWUA. 2015. V. 6. №2. P. 78-97. DOI:10.22667/JOWUA.2015.06.31.078

References:

1. Markin, D. O., Komashinsky, V. V., & Baranov, I. Yu. (2015). Mobile Device Security Profile Management Model Using Access to Services With Different Privacy Level. *Information Technology, 21*(8). 611-618.

2. Mobile Top 10 2016-Top 10 https://clck.ru/Gsqgp.

3. Zubkov, K. N., & Diasamidze, S. V. (2017). Formation Security Problems in Applications for Mobile Systems. *Intellectual Technologies on Transport,* (2). 40-46.

4. Lu, L., Li, Z., Wu, Z., Lee, W., & Jiang, G. (2012). Chex: statically vetting android apps for component hijacking vulnerabilities. *In Proceedings of the 2012 ACM conference on Computer and communications security. ACM.* 229-240. doi:10.1145/2382196.2382223

5. Android Security Bulletin - September 2017. https://clck.ru/GsqhF

6. Android Security Bulletin - May 2018. https://clck.ru/GsqhQ

7. Android Security Bulletin - January 2018. https://clck.ru/Gsqhg

8. Android Security Bulletin - April 2018. https://clck.ru/Gsqhx

9. Android Security Bulletin - November 2016. https://clck.ru/GsqiK

10. Shabtai, A., Tenenboim-Chekina, L., Mimran, D., et al. (2014). Mobile Malware Detection through Analysis of Deviations in Application Network Behavior. *Computers & Security, 43*.1-18. DOI:10.1016/j.cose.2014.02.009

11. Aleksandrov, I. A., Safin, K. L., Troshina, K. N., & Chernov, A. V. (2016). Static binary analysis of mobile applications for the android platform, according to the requirements of information security. *Moscow University Computational Mathematics and Cybernetics,* (3). 44-49.

12. Enck, W., Ongtang, M., & McDaniel, P. (2009). On Lightweight Mobile Phone Application Certification. *Proceedings of the 16th ACM Conference on Computer and Communications Security, NY, USA: ACM.* 235-245. DOI:10.1145/1653662.1653691

13. Tsyganenko, N. P. (2015). The Static Analysis of Mobile applications code as Vulnerabilities detection Method. *Proceedings of BSTU. Series 3: Physical and mathematical Sciences and Informatics*, *6*(179). 200-203.

14. Skovoroda, A., & Gamayunov, D. (2015). Securing mobile devices: malware mitigation methods. *JoWUA, 6*(2), 78-97.DOI:10.22667/JOWUA.2015.06.31.078

Работа поступила
в редакцию 20.06.2019 г.

Принята к публикации
24.06.2019 г.

Ссылка для цитирования:

Воронин А. А., Виноградов Д. В., Воронина А. А. Оценка защищенности среды выполнения операционной системы Android // Бюллетень науки и практики. 2019. Т. 5. №7. С. 154-161. https://doi.org/10.33619/2414-2948/44/20

Cite as (APA):

Voronin, A., Vinogradov, D., & Voronina, A. (2019). Security Estimation of Android Execution Environment. *Bulletin of Science and Practice, 5*(7), 154-161. https://doi.org/10.33619/2414-2948/44/20 (in Russian).